

AMENDMENTS TO THE CLAIMS

This listing of claims replaces all prior versions, and listings, of claims in the application:

Listing of Claims:

1. (Currently Amended) A computer-implemented method for allowing a software application to run using a specified version of one or more shared assemblies, wherein the specified version of the one or more shared assemblies used in the application is not compiled in the executable files of the application, the method comprising:

~~receiving a request from an application to load an assembly, the request not including assembly version data, including when the assembly is among a plurality of assemblies located in a same directory;~~

accessing a manifest, the manifest comprising a software data file including metadata describing applications' dependencies on assembly versions, the manifest being stored in non-volatile storage and being associated with an application that is configured to load specified versions of one or more assemblies upon initiation;

~~building an activation context based on a the accessed manifest, the manifest being associated with the application that requested the loading of the assembly, wherein the activation context comprises a table of contents that maps global, version independent names to a particular specified version of at least one of the one or more assemblies that are indicated in the manifest an assembly, and distinguishes between versions of assemblies based on the version indicated by the manifest, the activation context being associated with the application in response to the application's request to load an assembly;~~

loading the activation context into computer system memory in a persistable binary form to provide accelerated access to the activation context;

receiving a request from an application to load an assembly from among a plurality of assemblies located in a same directory;

~~consulting information in a the activation context manifest associated with the application, the manifest being stored separately from the application and any changes made to the manifest being implemented without having to recompile the manifest, the manifest being used to identify a particular specified version of the requested assembly~~

without referring to the received manifest, in response to receiving the request to load the assembly the building of the activation context; and

providing the particular specified version of the assembly for use by the application.

2. (Original) The method of claim 1, wherein the request corresponds to a request to load a privatized assembly.

3. (Original) The method of claim 1, wherein the request corresponds to a request to load a shared assembly.

4. (Original) The method of claim 3, wherein the shared assembly is maintained in an assembly cache.

5. (Currently Amended) The method of claim 1, wherein consulting information ~~associated with the application~~ in the activation context to ~~determine~~ identify a particular version of the assembly includes searching for a mapping from a version independent name provided by the application to a version specific assembly.

6. (Previously Presented) The method of claim 5, wherein no mapping from the version independent name to a version specific assembly is present, and wherein providing the particular version of the assembly for use by the application comprises providing a default version.

7. (Original) The method of claim 1, wherein providing the particular version of the assembly comprises accessing a file corresponding to the assembly and loading the assembly into memory from the file.

8. (Currently Amended) The method of claim 1, wherein the information ~~associated with the application~~ in the activation context includes a mapping between a version independent name provided by the application and a version specific file system path and filename of the

particular version of the assembly, and wherein providing the particular version of the assembly comprises returning the path and filename to an assembly loading mechanism.

9. (Previously Presented) The method of claim 8, wherein the application is stored as an application executable file in a folder, and wherein the version of the assembly is stored as another file in the same folder.

10. (Original) The method of claim 8, wherein the filename corresponds to a file in an assembly cache.

11. (Currently Amended) The method of claim 1, wherein the information ~~associated with the application~~ in the activation context is derived from application manifest.

12. (Currently Amended) The method of claim 11, wherein the information ~~associated with the application~~ in the activation context is further derived from at least one assembly manifest.

13. (Currently Amended) The method of claim 1, wherein the information ~~associated with the application~~ in the activation context is constructed during a pre-execution initialization phase.

14. (Currently Amended) The method of claim 1, wherein the information ~~associated with the application~~ in the activation context is persisted into a non-volatile memory.

15. (Previously Presented) A computer-readable storage medium having computer-executable instructions for performing the method of claim 1.

16. (Currently Amended) A computer-implemented method for allowing a software application to run using a specified version of one or more shared assemblies, wherein the specified version of the one or more shared assemblies used in the application is not compiled in the executable files of the application, the method comprising:

accessing a manifest, the manifest comprising a software data file including metadata describing applications' dependencies on assembly versions, the manifest being stored in non-volatile storage and being associated with an application that is configured to load specified versions of one or more assemblies upon initiation

building an activation context based on a~~the~~ accessed manifest, the manifest being associated with the application that requested the loading of the assembly, wherein the activation context comprises a table of contents that maps global, version independent names to a particular specified version of at least one of the one or more assemblies that are indicated in the manifest an assembly, and distinguishes between versions of assemblies based on the version indicated by the manifest, the activation context being associated with the application, the activation context identifying dependency information;

loading the activation context into computer system memory in a persistable binary form to provide accelerated access to the activation context;

receiving a request from an application to load an assembly from among a plurality of assemblies located in a same directory;

interpreting the dependency information associated with the application, the dependency information identifying at least one particular specified version of an assembly without referring to the received manifest, in response to the request to load the assembly including when the assembly is among a plurality of assemblies having at least some components located in a same directory; and

associating with the application at least one mapping based on the dependency information, each mapping relating a version independent assembly name that the application may provide to a version specific assembly identified in the dependency information.

17. (Previously Presented) The method of claim 16, wherein the dependency information is provided in an application manifest associated with the application.

18. (Previously Presented) The method of claim 17, wherein the application manifest is associated with the application by being stored in a common folder with an application executable file that corresponds to the application.

19. (Previously Presented) The method of claim 16, wherein at least one mapping maps a version independent name to an assembly stored in a common folder with an application executable file that corresponds to the application.

20. (Original) The method of claim 16, wherein at least one mapping maps a version independent name to a shared assembly in an assembly cache.

21. (Previously Presented) The method of claim 16, wherein the dependency information provided by the application corresponds to an assembly having an assembly manifest associated therewith, and further comprising, interpreting the assembly manifest.

22. (Original) The method of claim 21, wherein the assembly manifest specifies that a particular version of an assembly be replaced with another version of that assembly.

23. (Original) The method of claim 21, wherein the assembly manifest specifies at least one particular version of another assembly on which the assembly having an assembly manifest is dependent.

24. (Previously Presented) The method of claim 16, wherein the dependency information is interpreted in response to receiving a request to execute the application.

25. (Original) The method of claim 16, wherein the at least one mapping is maintained in an activation context, and further comprising, persisting the activation context.

26. (Previously Presented) The method of claim 25, wherein associating with the application the at least one mapping comprises retrieving a persisted activation context.

27. (Previously Presented) The method of claim 25, wherein associating with the application the at least one mapping comprises constructing a new activation context.

28. (Original) The method of claim 27, wherein the new activation context is constructed upon determining that an activation context does not exist.

29. (Original) The method of claim 27, wherein the new activation context is constructed upon determining that an existing activation may not be not coherent with current policy.

30. (Previously Presented) The method of claim 16, further comprising, running the application, receiving a request from the application to load an assembly, the request including data corresponding to a version independent name of the assembly and providing a particular version of the assembly for use by the application based on a mapping therefor.

31. (Previously Presented) A computer-readable storage medium having computer-executable instructions for performing the method of claim 16.

32-41. (Cancelled)

42. (Currently Amended) A system in a computing environment, comprising:

an accessing mechanism configured to access a manifest, the manifest comprising a software data file including metadata describing applications' dependencies on assembly versions, the manifest being stored in non-volatile storage and being associated with an application that is configured to load specified versions of one or more assemblies upon initiation

a building mechanism configured to build an activation context based on the accessed manifest, wherein the activation context comprises a table of contents that maps global, version independent names to a specified version of at least one of the one or more assemblies that are indicated in the manifest, the activation context identifying dependency information;

a loading mechanism configured to load the activation context into computer system memory in a persistable binary form to provide accelerated access to the activation context;

a receiving mechanism configured to receive a request from an application to load an assembly from among a plurality of assemblies located in a same directory;

an initialization interpretation mechanism configured to interpret dependency data associated with an-the application, the dependency data corresponding to at least one specified assembly version on which the application depends without referring to the received manifest, in response to the request to load the assembly, each assembly version corresponding to an assembly having version information associated therewith and contained in a directory structure among a plurality of assemblies;

an activation context based on a manifest, the manifest being associated with an application, wherein the activation context maps global, version independent names to a particular version of an assembly, and distinguishes between versions of assemblies based on the version indicated by the manifest, the activation context associated with the application and constructed by the initialization mechanism based on the dependency data, the activation context relating at least one version independent assembly identifier provided by the application to a version specific assembly; and

an associating mechanism configured to associate with the application at least one mapping based on the dependency information, each mapping relating a version independent assembly name that the application may provide to a version specific assembly identified in the dependency information; and

a version matching mechanism configured to access the activation context to relate a version independent request from the application to a version specific assembly.

43. (Previously Presented) The system of claim 42, wherein the dependency data is included in the application manifest.

44. (Original) The system of claim 42, wherein the dependency data is included in an XML file.

45. (Original) The system of claim 42, wherein the initialization mechanism persists the activation context.

46. (Previously Presented) The system of claim 42, further comprising, an assembly loading mechanism configured to communicate with the application and the version matching mechanism to load the version specific assembly upon a request by the application to load a requested assembly, wherein the request does not include version specific data.

47. (Original) The system of claim 46, wherein the assembly loading mechanism loads the version specific assembly from an assembly cache.

48. (Original) The system of claim 42, wherein the dependency data identifies an assembly that has assembly dependency data associated therewith, the assembly dependency data corresponding to at least one other assembly version on which the assembly depends, and wherein the initialization mechanism adds information that corresponds to the assembly dependency data to the activation context.

49. (Previously Presented) A computer-readable storage medium having computer-executable modules configured to implement the system of claim 42.